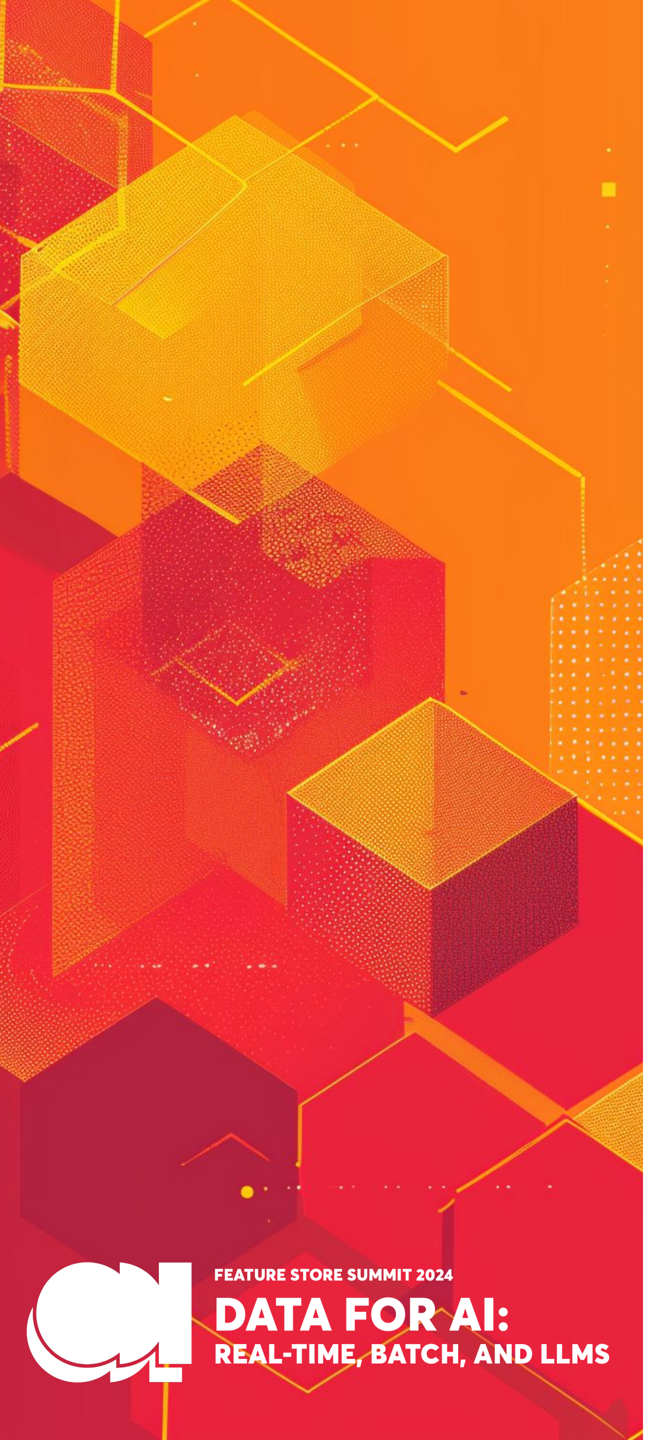


Embeddings @ Uber

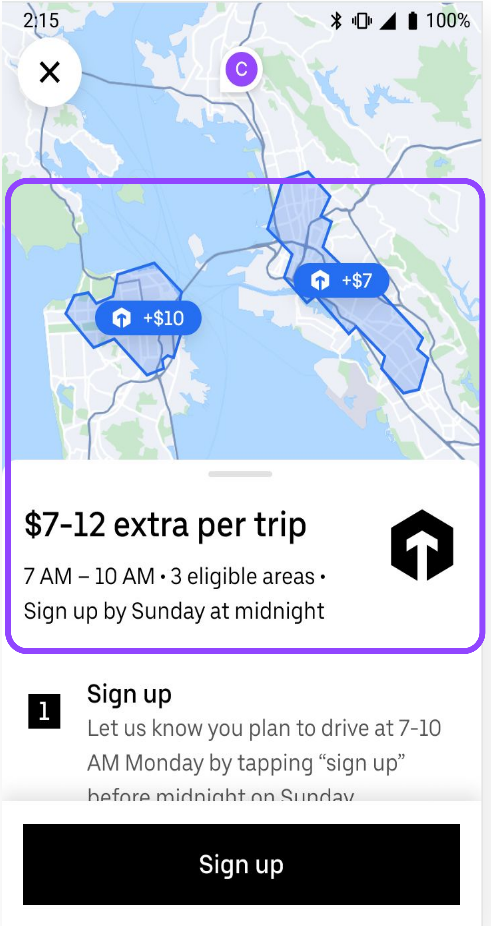
Dhruva Dixith Kurra, Engineer, Uber

Organized by HOPSWORKS

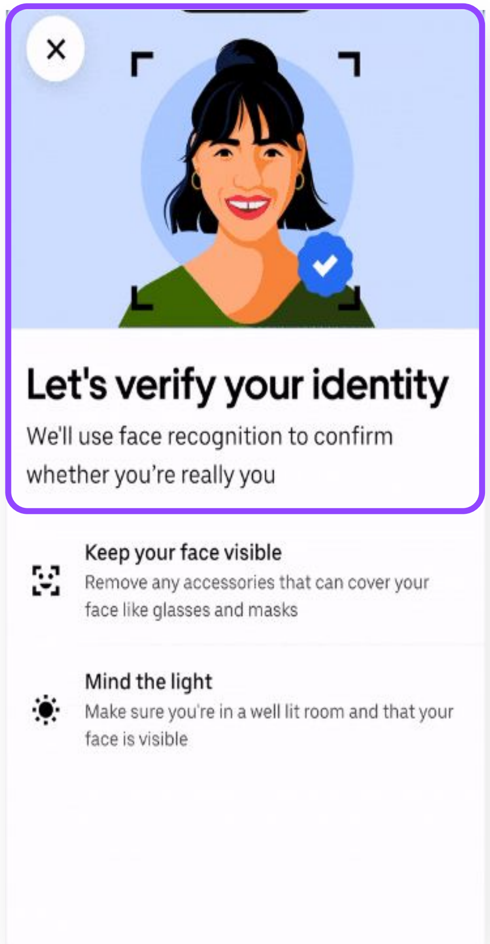
You will learn:

- ML at Uber
- Embeddings: What Are They Anyway…?
- Two Tower Architecture
- Training Challenges & Solutions
- Architecture
- Impact

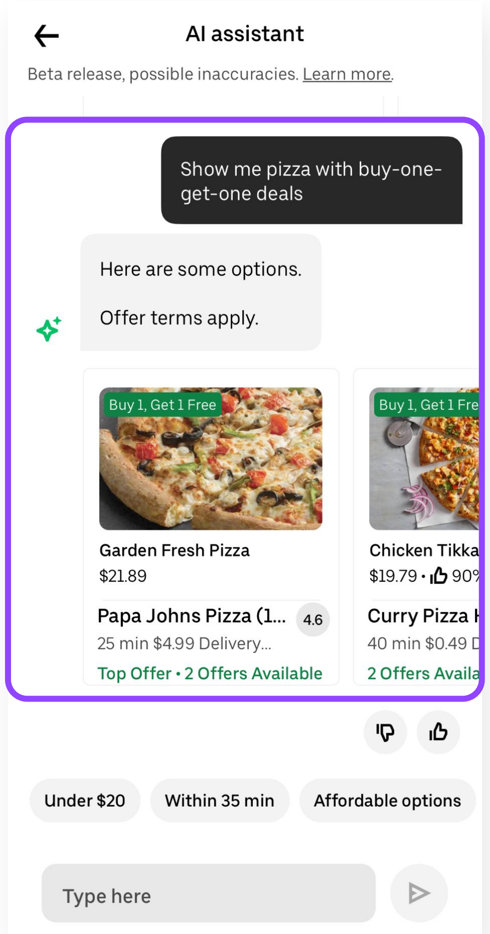
Context | Problem | Solution | Embeddings | TTE | Architecture | Impact



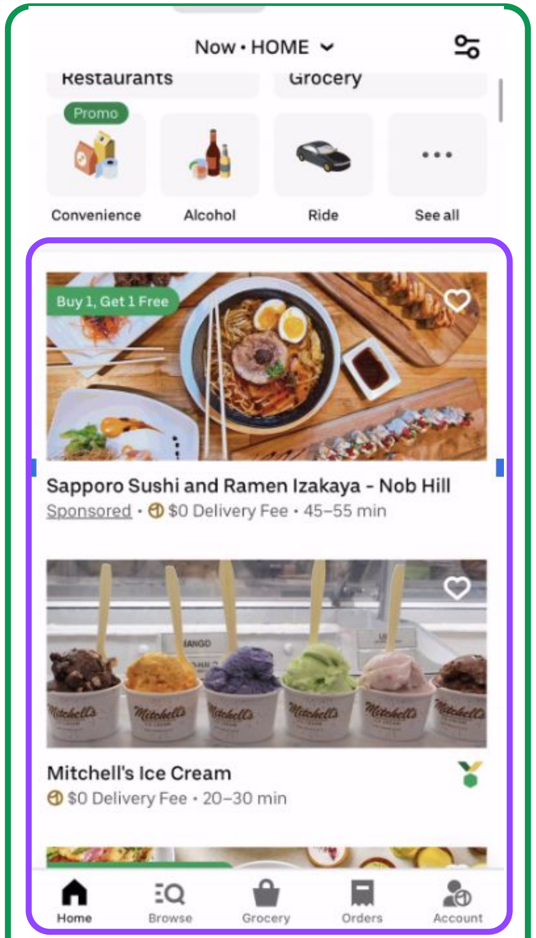
Efficient Marketplace
Matching, Routing, Dispatch, Pricing, Incentives



Risk & Safety
ID verification, Fraud & Incident prevention



Chatbots
Customer support, assistants, co-pilots

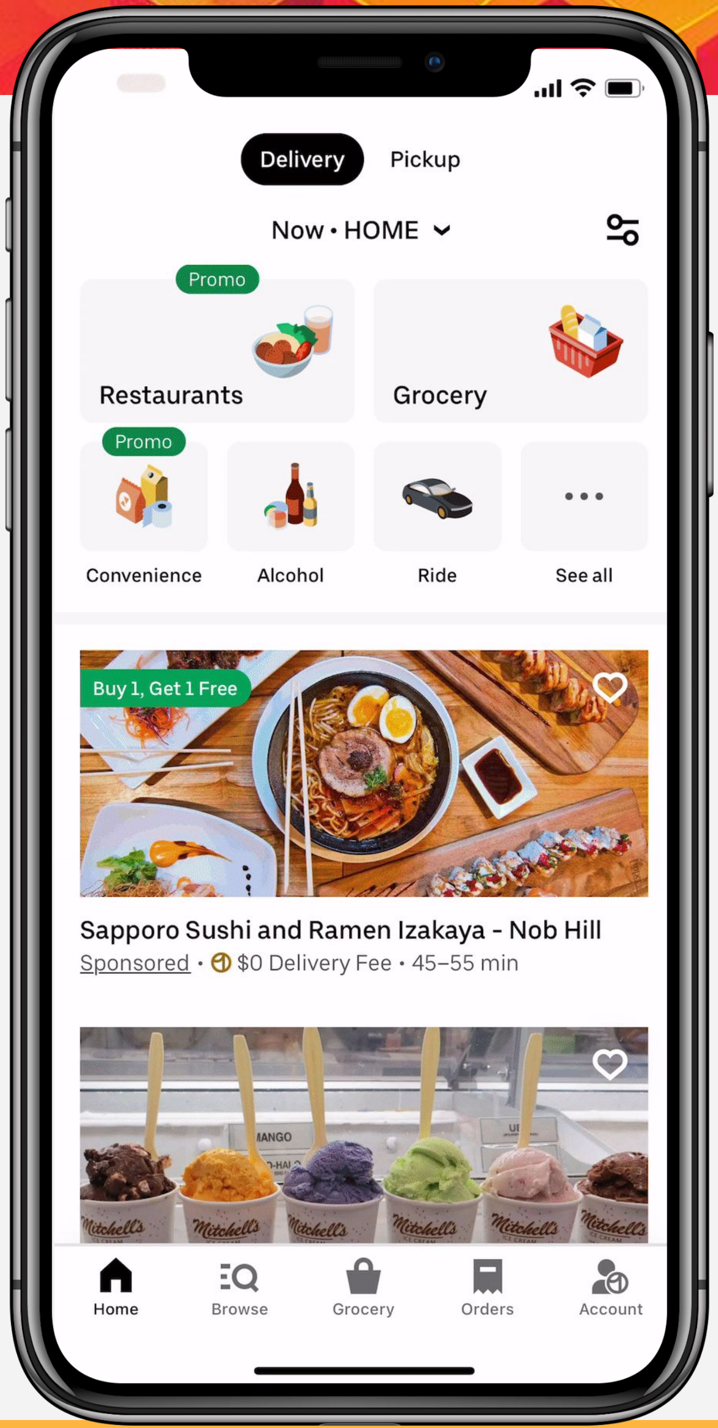


Personalization & Recommendation
Eats Feed, Search, Rides

Eats Homefeed ranking has a direct impact on users:

- ➔ **95%** of users start their journey with Home Feed
- ➔ **Majority** of all orders originated on Home Feed

However, we have minimum real estate & time to convert



Existing solution

⚠️ Lower performance
⚠️ High computing costs
⚠️ Scaling blockers

Problem 1: Lack of efficient retrieval model

We needed to retrieve the best stores out of thousands in just 50ms.

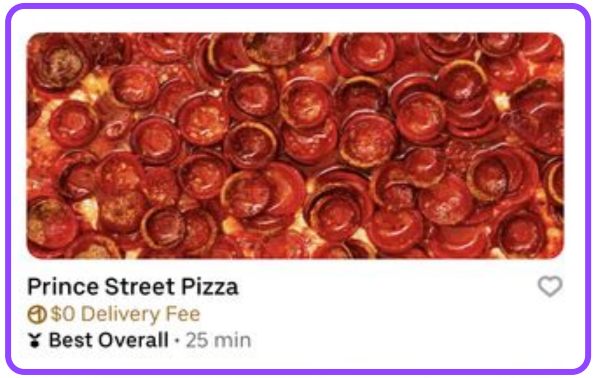
Problem 2: Existing technology could not scale (Deep Matrix Factorization (DeepMF)).

➔ Required 1,000+ city models globally (location based)
➔ Not reusable
➔ Very expensive to maintain (200,000 CPU hours per week -> continued to increase with # of cities)

Context | Problem | **Solution** | Embeddings | TTE | Architecture | Impact

⭐ Long term solution = Embeddings ⭐

✅ **Better performance**
✅ **Lower computing costs**
✅ **Eliminated scaling blockers**



Champion use case = Eats Homefeed

➔ Proves the value and replaced DeepMF.

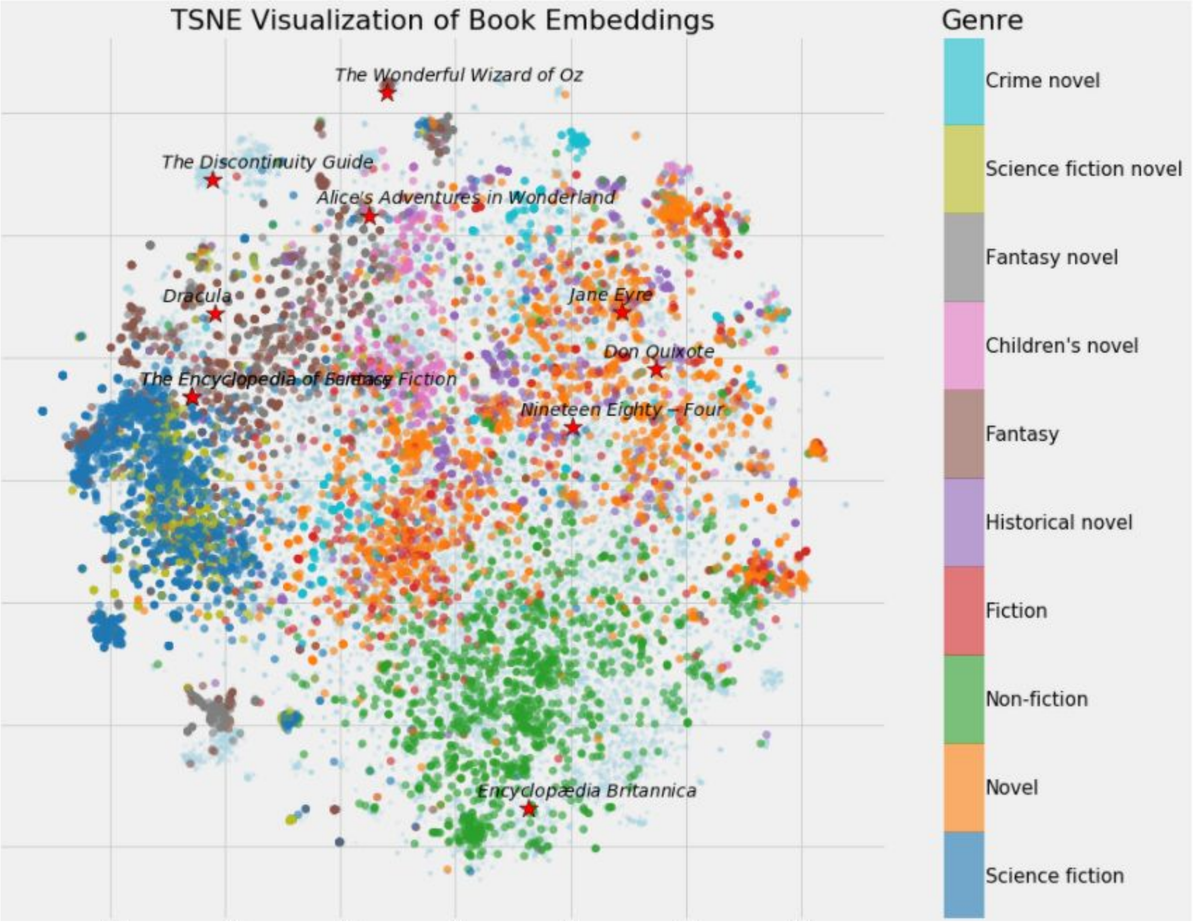
➔ Retrieves personalized stores in 50ms, enabling customers to quickly & easily find items by selecting the best store for them.

➔ We brought embeddings to Uber by building the platform capability so they can be scaled, reused, and transferred beyond our initial use case.

<takeaway Embeddings condense down all features into a single vector>

Embeddings are a type of feature for modern AI:

- Highly condensed vectors (<example>)
- Generally work for any entity such as eater, store, item, rider, location
- More natural for AL/ML tasks such as clustering

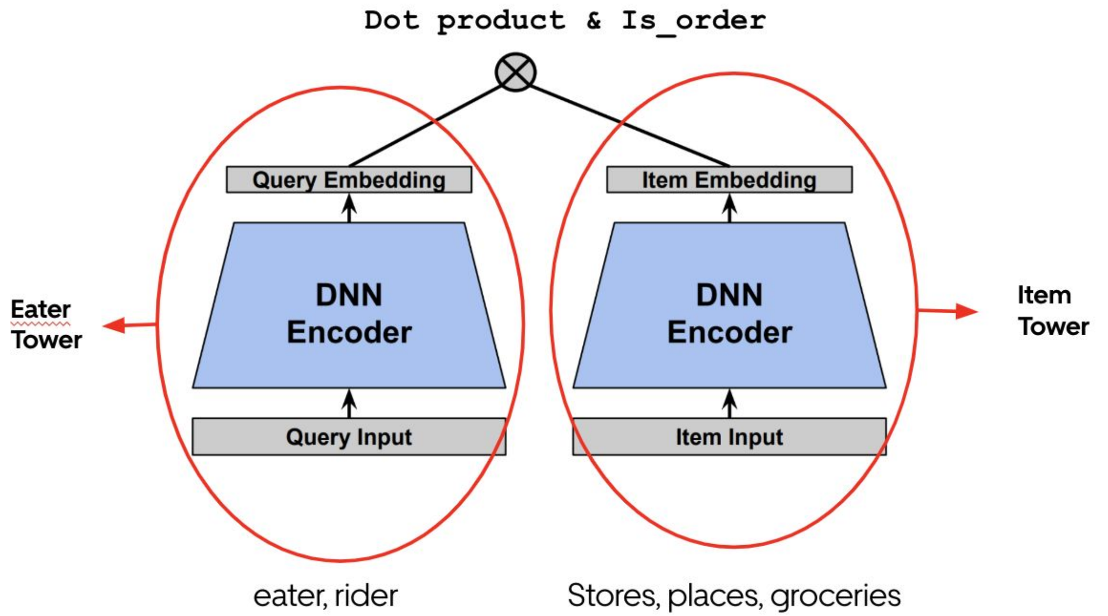


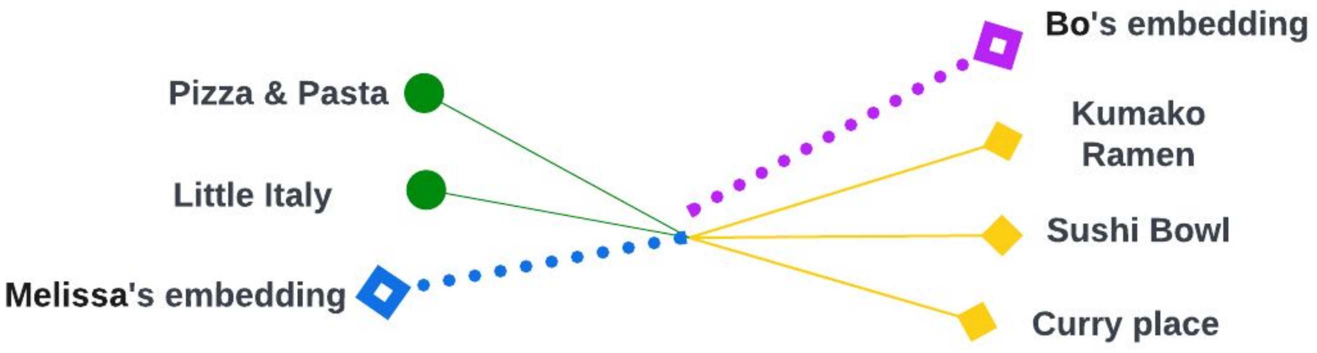
Source: [Neural Network Embeddings Explained] | Author: Will Koehrsen

We needed a large scale Embedding system to Generate Embeddings for several entities at Uber

➔ Two Tower Model:

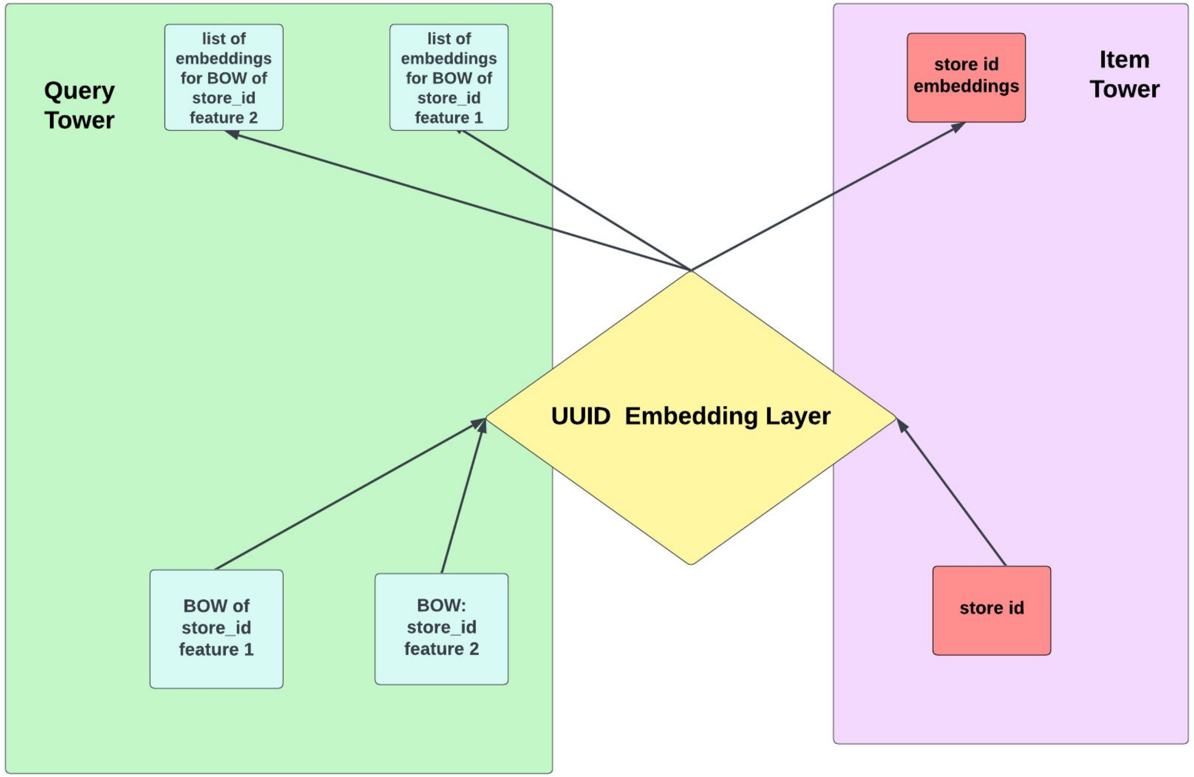
- A special way to learn embeddings via user behavior such as click and order
- Eater tower: generates embeddings for eater, rider offline or realtime
- Item tower: generates embeddings for store, grocery and place offline
- Best AI solution for retrieval stage in recommendation system





How to handle large cardinality

- Hundreds of millions eaters
  - Eater_id as a feature = <u>huge model 99% of the model size</u> 😦
  - Cold start problem for new users & hash collision
  - Data can be flawed (i.e. incorrect cuisine tags)
- Millions Stores
  - Past engagements with stores as a proxy of the user_id
  - Layer sharing between store and eater tower
    - Transformers learn eater's list of engagement
- Resulting in:
  - <u>100x model size drop!!</u> 😃



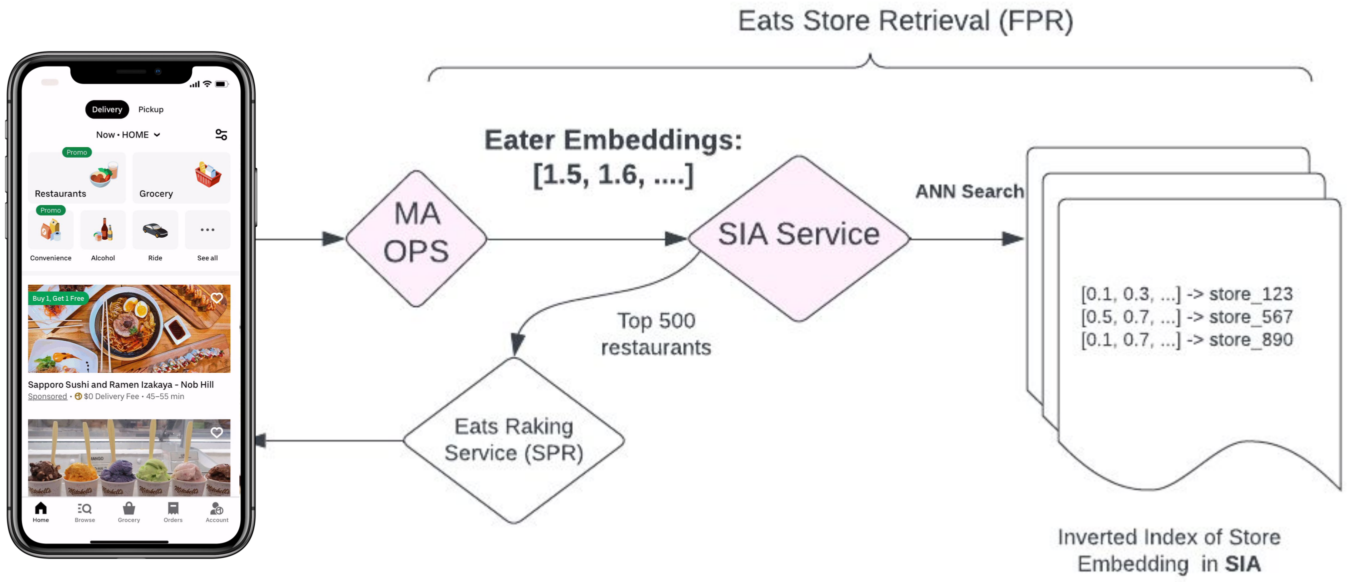
How to handle location based model

- **Uber recommendation system is very location centric**
  - Baseline model: city-wise deep-mf for restaurant retrieval
- **Spatial indexing: geo base problem**
  - find all available restaurants around eater by geolocation distance in real time (boundary control)
  - sort train data by geo-hash so that eaters and close restaurants are in the same batch
- **LogQ Correction for In-batch Negatives**
  - Create sufficient negatives using in-batch negatives: 4k to 8K
  - Down-sample restaurants in a batch with LogQ:
  - Q is sampling probability in a batch, w is the item weights in the whole data
-

*Includes mechanisms to bound time based constraints (5pm can deliver 5miles; 10pm can deliver 10miles), allowing only available items.

Overall Flow:

- ➔ Generate item/store embeddings and eater model in our online prediction service
- ➔ Index Item/Store embeddings in our retrieval/search engine
- ➔ Eater embedding computed from prediction service at realtime
- ➔ Search engine scores and fetches the most relevant stores/items for eater.



*Includes mechanisms to bound the problem through time based constraints (5pm can deliver 5miles; 10pm can deliver 10miles), allowing only available items.

Embeddings as a Feature

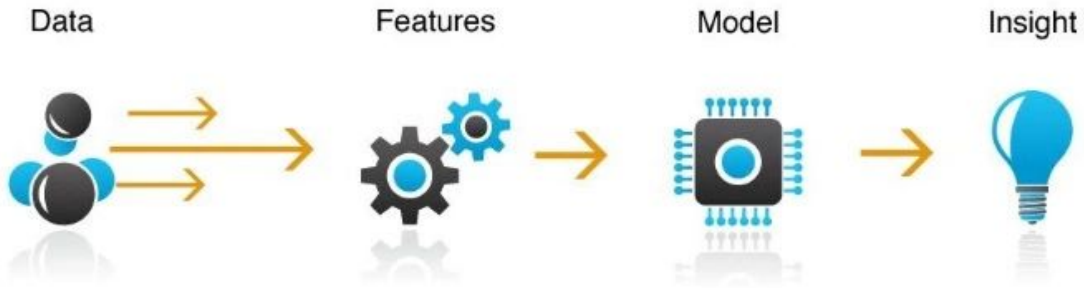
Feature: data with predictive power that is used as input for models to make predictions.

Palette: the industry's first feature store (2017).

One-stop shop for feature engineering needs

- ➔ Stores & manages features
- ➔ Serves feature data consistently for training & inference

Our feature Store powers Uber's business

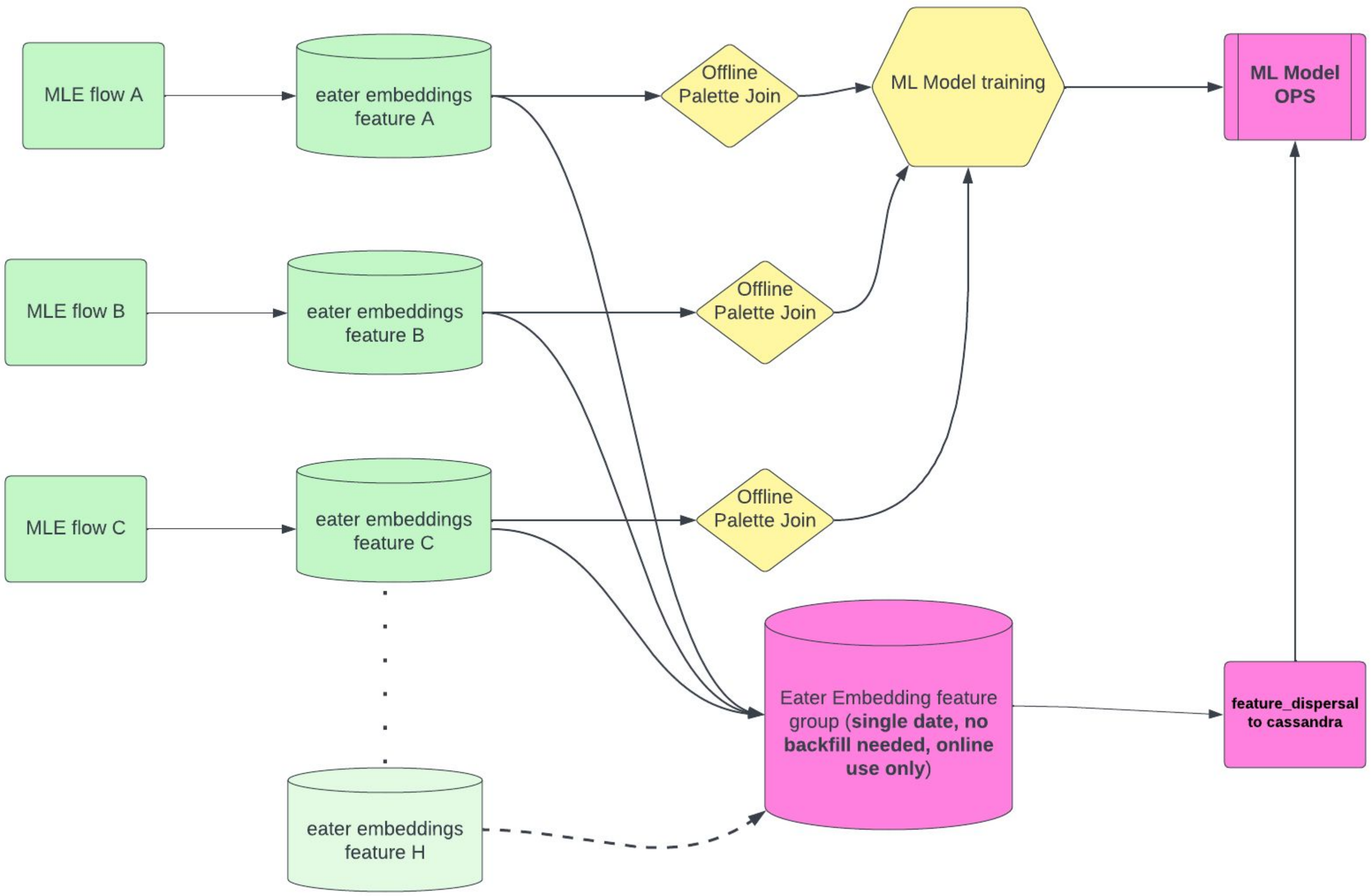


1. Embedding as a feature in **another** Tier 1-2 model such as DeepCVR, DeepNI (**transferability**)

2. Managed and used via palette service (**servability**)

3. Feature transformation and modeling can be **standardized** (data type is array of double)

4. Feature exploration and engineering is simpler (one embeddings cover tens or hundreds of other regular features)



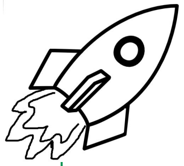
V2.0 Global

Single global model, eliminating thousands of individual models from DeepMF

Results

➔ top 20th percentile for improving mainfeed CVR, and homefeed CVR

	Recall@100	Recall@200	Recall@300	Recall@400	Recall#500
Deep MF	0.4858	0.634	0.7198	0.7766	0.8165
TTE	**0.7856**	**0.8742**	**0.9139**	**0.9364**	**0.9506**



Infra wins

➔ Single global model replaces **thousands** city DeepMF models
➔ Scalable to **millions** of different types of users and multiple trips and sessions
➔ Decreased model training from **200,000** to **4800 core hours** **per week**

Acknowledgements

It took a village to make it a reality, special shout out to:

Bo Ling , **Chun Zh** , **Nicholas Marcott** , **Eric Chen** , **Melissa Barr** and the Michelangelo (ML Platform) team at Uber.

Questions?